Lab Report: HCS12 Waveform Generation and Analysis

Objective: (report is an extension of homework 10)

The objective of this assignment was to implement and analyze the generation of various periodic waveforms (Sawtooth, Triangle, Square) at different frequencies using the HCS12 microcontroller's Timer Output Compare interrupt mechanism. The generated waveforms were outputted via the Serial Communication Interface (SCI) and subsequently visualized and analyzed using a Python script.

Methodology:

1. **HCS12 Assembly Program (`cmpen472hw11_Mbatia.asm`)**:

1.1. A real-time clock and command interface were implemented.

1.2. Timer Channel 6 Output Compare (OC6) was configured to generate interrupts at a fixed interval of 125µs (corresponding to an 8kHz sampling rate, as $\frac{1}{125\mu S} = 8000\ Hz1$). This interval was achieved by setting `TC6H` with an offset of 3000 counts relative to the timer counter `TCNTH`, assuming a 24MHz bus clock (`TSCR2` configured for TCLK = BCLK/1).

1.3. The Timer OC6 Interrupt Service Routine (`oc6isr`) was responsible for calculating the next data point of the selected waveform.

1.4. Five distinct waveform types were implemented, selected via commands (`gw`, `gt`, `gq`, `gw2`, `gq2`):

1.4.1. `WAVE_SAW` (Sawtooth): Linearly increments from 0 to 255 over 256 samples (period = 32ms, frequency ≈ 31.25 Hz).

1.4.2. `WAVE_TRI` (Triangle): Linearly increments from 0 to 255 over 256 samples, then linearly decreases from 255 to 0 over 256 samples (period = 64ms, frequency ≈ 15.625 Hz).

1.4.3. `WAVE_SQUARE` (Square): Outputs 0 for 256 samples, then 255 for 256 samples (period = 64ms, frequency ≈ 15.625 Hz).

1.4.4. `WAVE_SAW_125` (Sawtooth 125Hz): Linearly increments from 0 to 252 (scaled 0-63 * 4) over 64 samples (period = 8ms, frequency = 125 Hz).

1.4.5. `WAVE_SQUARE_125` (Square 125Hz): Outputs 0 for 32 samples, then 255 for 32 samples (period = 8ms, frequency = 125 Hz).

1.5. Each waveform generation sequence produced exactly 2048 data points.

1.6. The calculated waveform data points were stored in `waveDataVal` and flagged (`waveDataReady`) for the main loop to print via the SCI port using the `pnum10` routine. The `RxData3Sim.txt` file captures this SCI output.

2. **Python Analysis Script (`plots.py`)**:

2.1. The script parses the `RxData3Sim.txt` file, identifying the start of each waveform sequence based on the specific command strings (`gw`, `gt`, `gq`, `gw2`, `gq2`).

2.2. It extracts the subsequent 2048 numerical data points corresponding to each triggered waveform.

2.3. Using `matplotlib.pyplot`, the script plots the time-domain representation of each full 2048-point waveform against a time axis calculated based on the 125µs sampling interval.

2.4. Using `numpy.fft`, the script calculates the Fast Fourier Transform (FFT) for each waveform to analyze its frequency components.

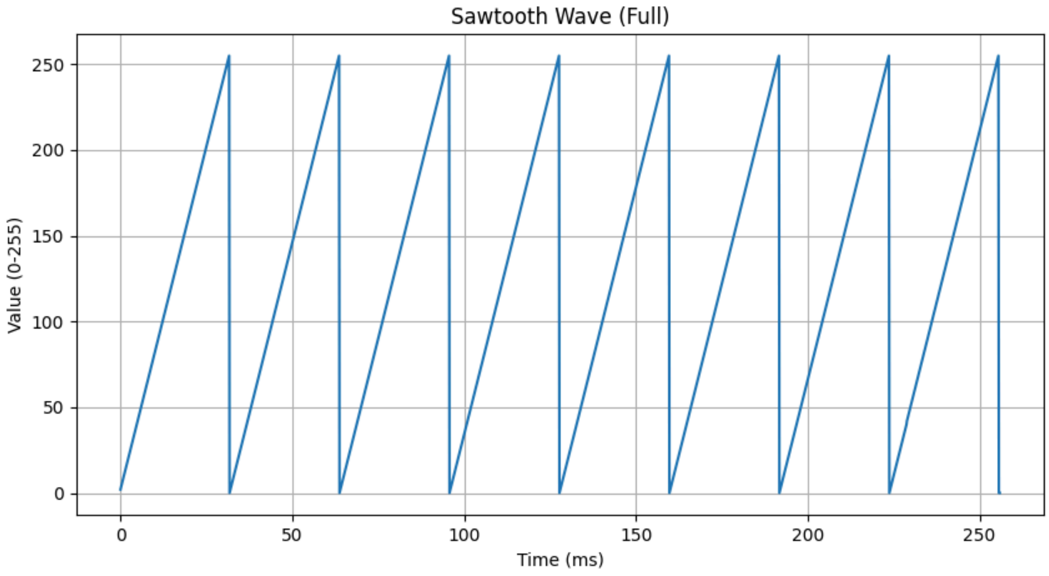
2.5. The magnitude spectrum (frequency vs. magnitude) is plotted for each waveform, focusing on the 0-2000 Hz range for clarity.

Results and Analysis:

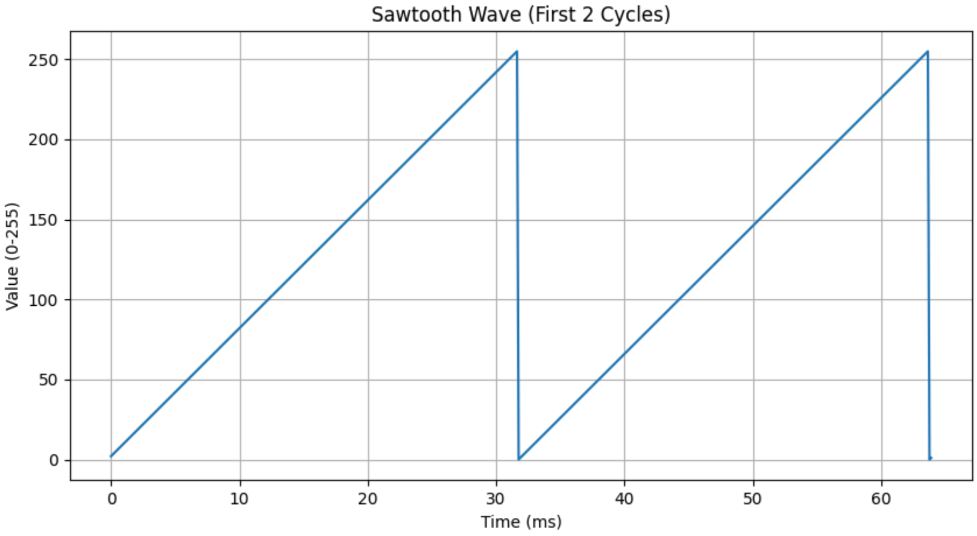
1. **Time-Domain Waveforms**:

1.1. Sawtooth Wave (Full):

1.1.1. The plot should show 8 complete cycles of a linear ramp from 0 to 255, each taking 256 samples (32ms).



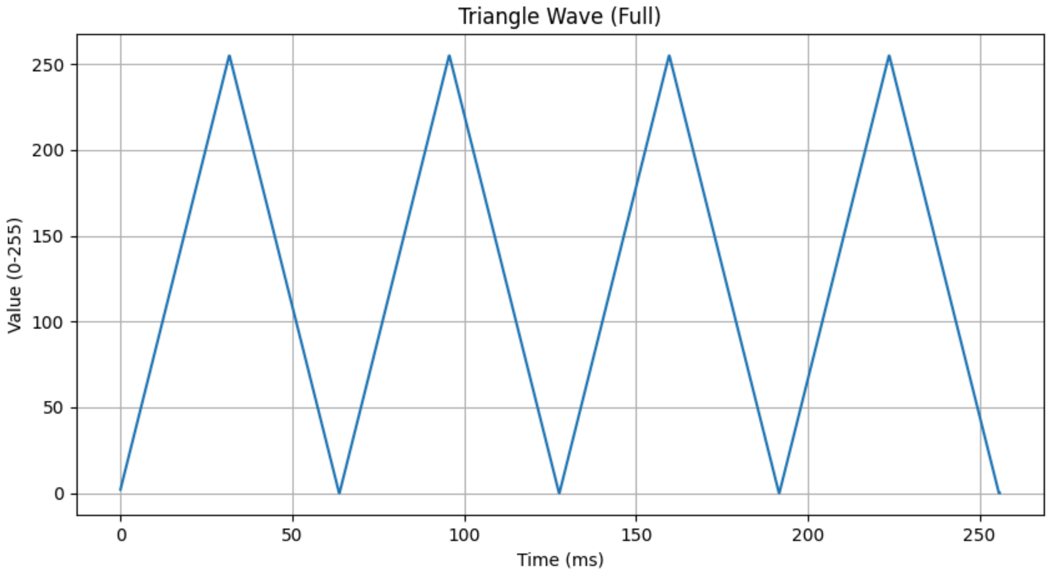
(Sawtooth Wave Full)



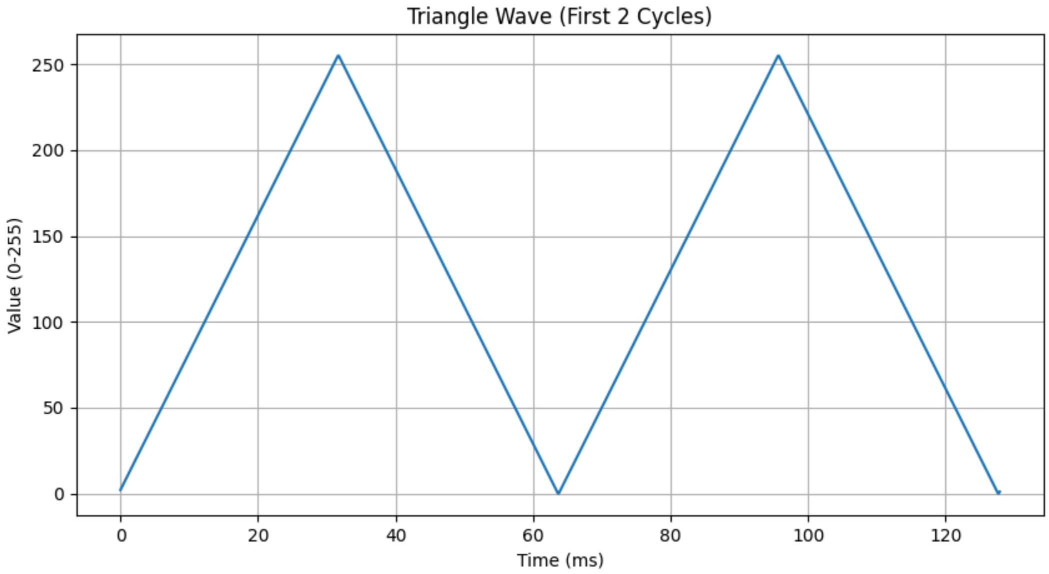
(Sawtooth Wave 2 cycles)

1.2. Triangle Wave (Full):

1.2.1. The plot should display 4 complete cycles. Each cycle consists of a linear ramp up from 0 to 255 (256 samples) followed by a linear ramp down from 255 to 0 (256 samples), for a total period of 512 samples (64ms).



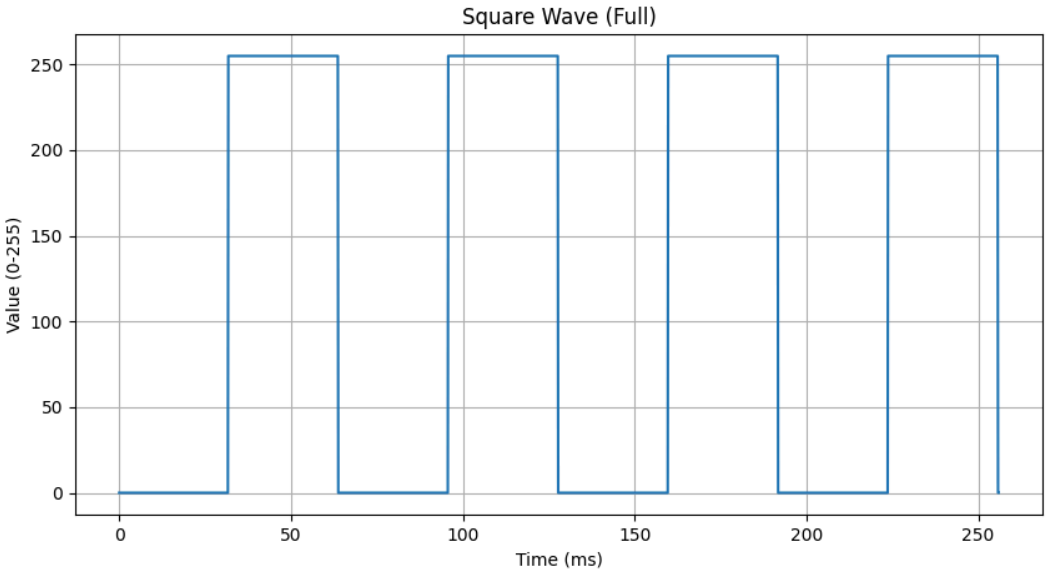
(Triangle Wave Full)



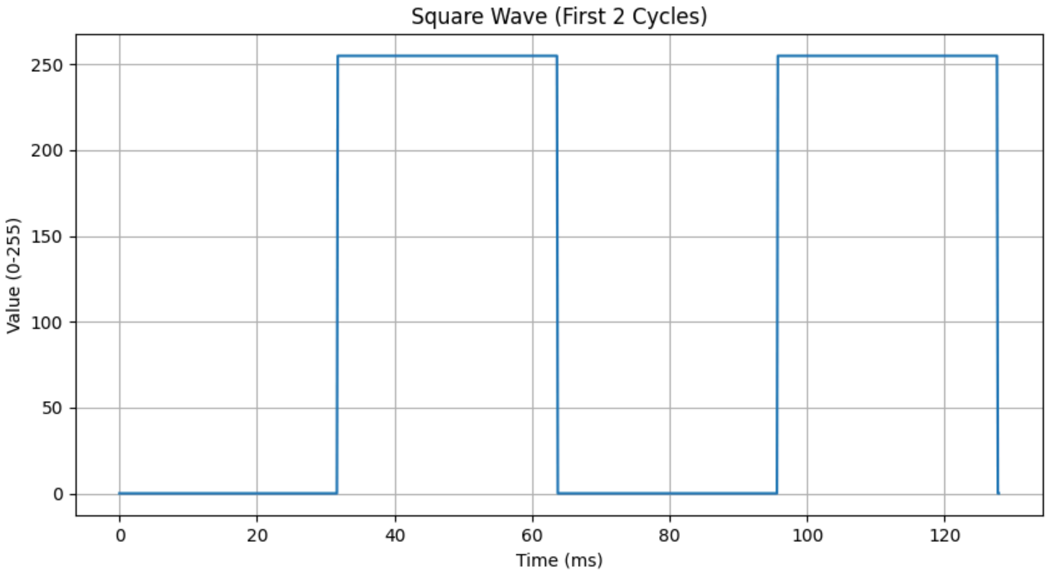
(Triangle Wave 2 Cycles)

1.3. Square Wave (Full):

1.3.1. The plot should show 4 complete cycles. Each cycle consists of a value of 0 for 256 samples followed by a value of 255 for 256 samples, with a total period of 512 samples (64ms).



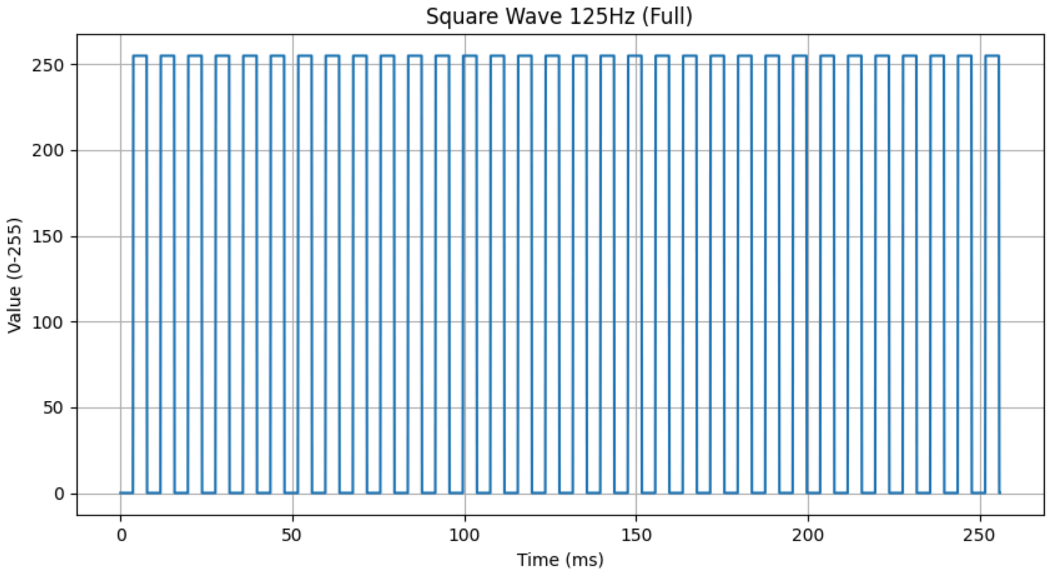
(Square Wave Full)



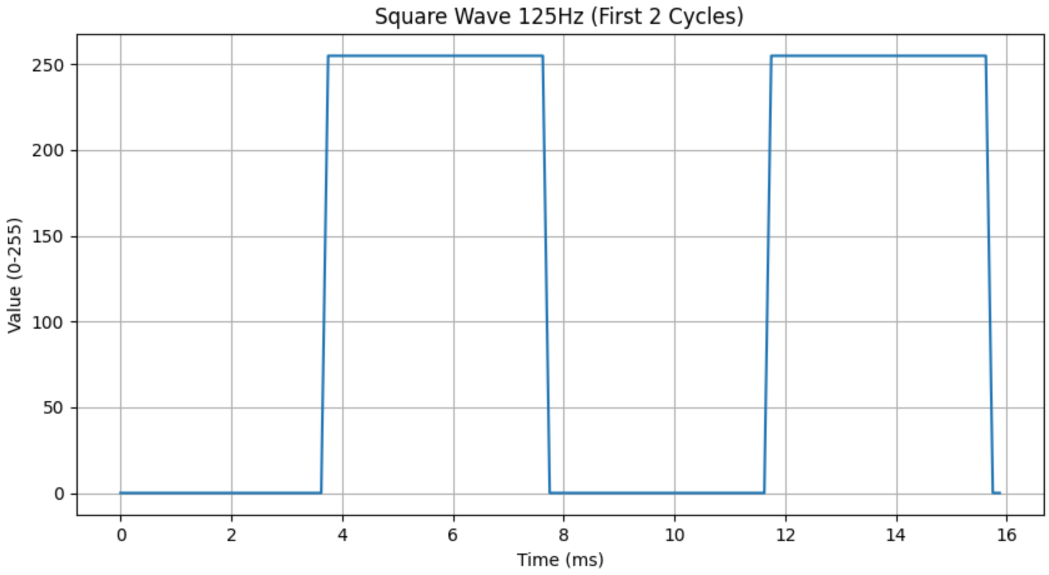
(Square Wave 2 Cycles)

1.4. Square Wave 125Hz (Full):

1.4.1. The plot should display 32 complete cycles. Each cycle consists of a value of 0 for 32 samples followed by a value of 255 for 32 samples, resulting in a period of 64 samples (8ms).



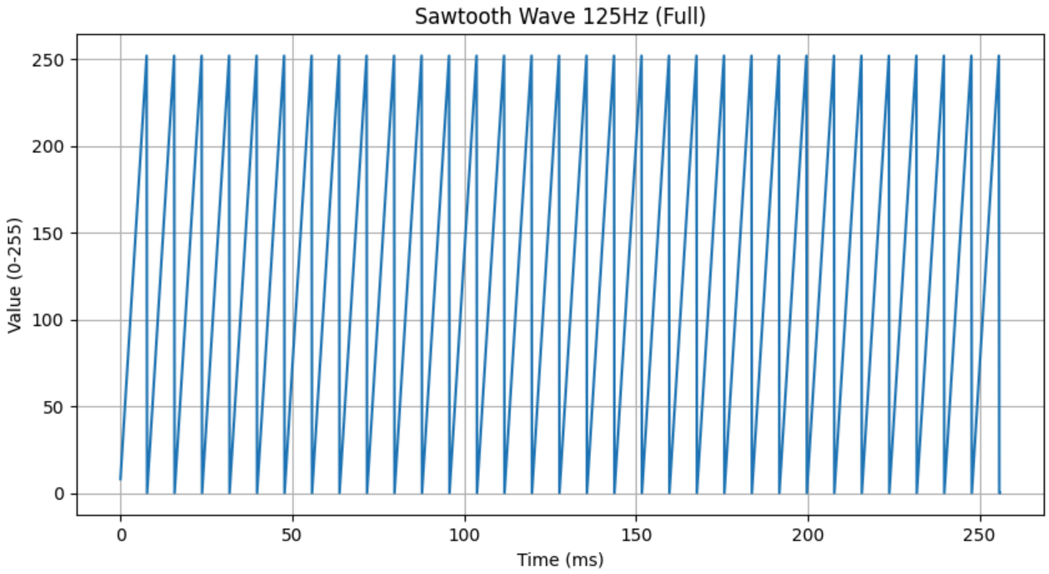
(Square Wave 125 Hz Full)



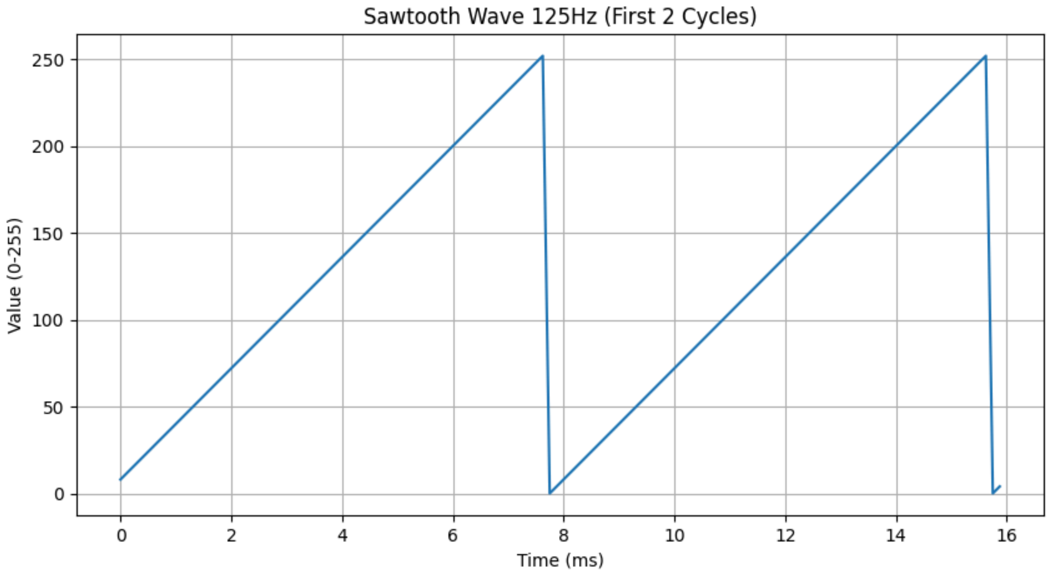
(Square Wave 125 Hz 2 Cycles)

1.5. Sawtooth Wave 125Hz (Full):

1.5.1. The plot should show 32 complete cycles of a linear ramp from 0 to approximately 252, each taking 64 samples (8ms).

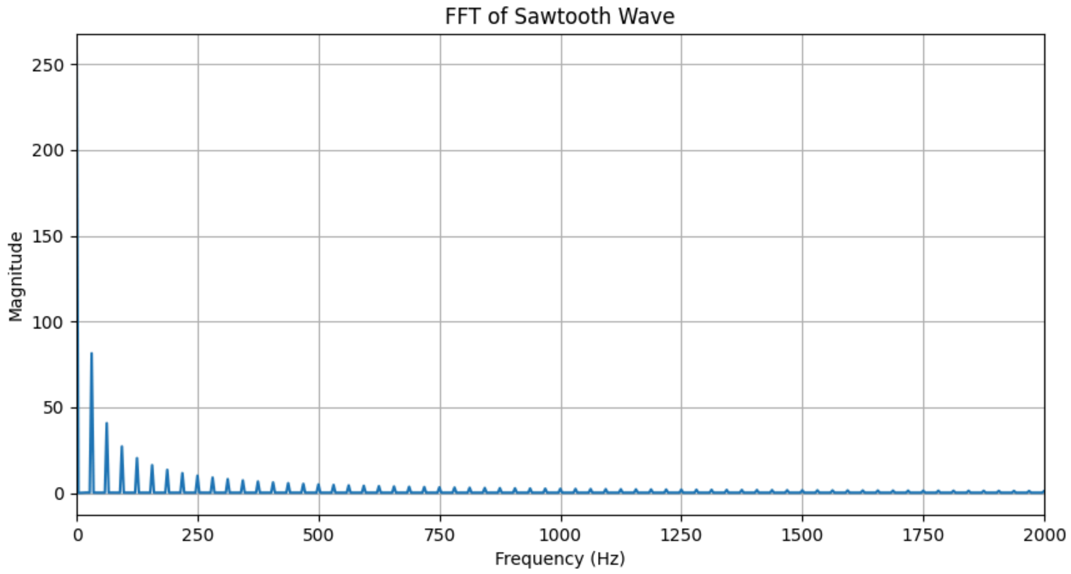


(Sawtooth Wave 125 Hz Full)



(Sawtooth Wave 125 Hz 2 cycles)

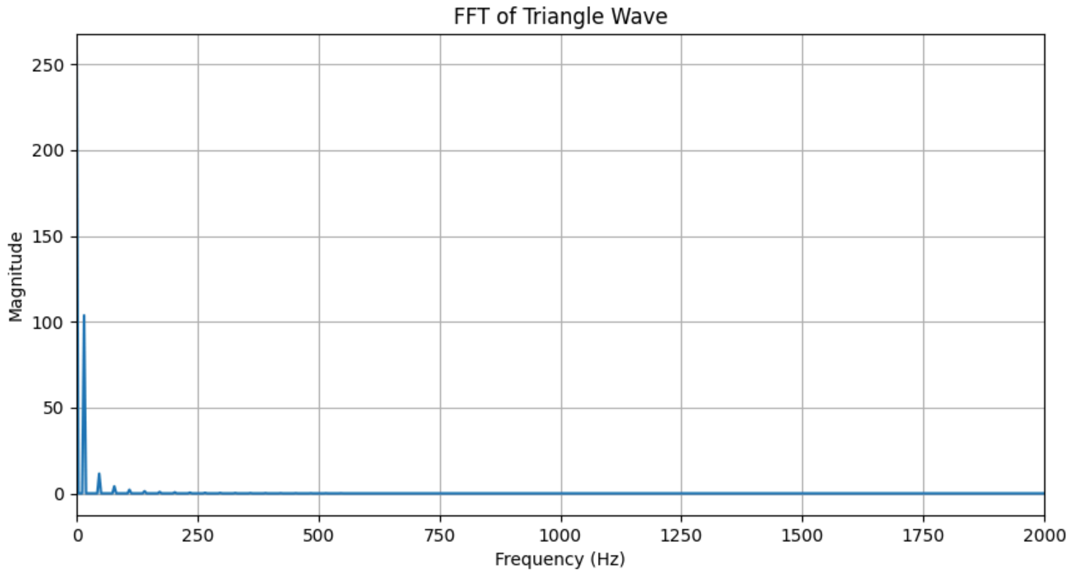
2. **Frequency-Domain Analysis (FFT)**:
    2.1. Sawtooth Wave:
        2.1.1. The FFT plot is expected to show a strong peak at the fundamental frequency (≈31.25 Hz) and significant peaks at integer multiples (harmonics) of this frequency. Both even and odd harmonics are present, with amplitudes generally decreasing as frequency increases.



(Sawtooth FFT)

2.2. Triangle Wave:

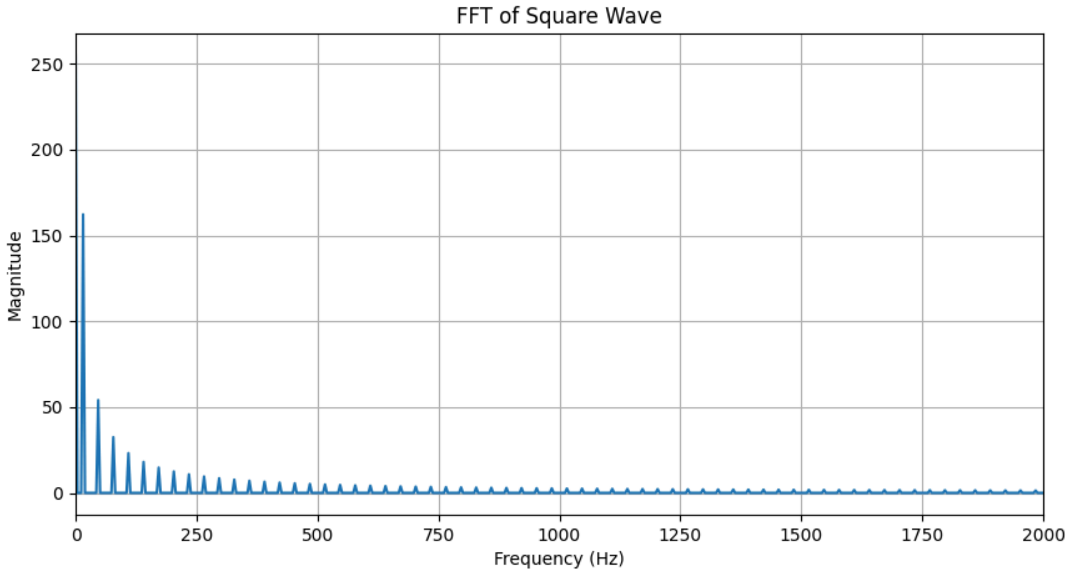
2.2.1. The FFT plot should show a strong peak at the fundamental frequency (≈15.625 Hz). Only odd harmonics (3rd, 5th, 7th, etc.) should be present with significant magnitude. The amplitude of these harmonics decreases more rapidly than for the sawtooth wave (proportional to $\frac{1}{n^2}$), where n is the harmonic number).



(Triangle FFT)

2.3. Square Wave:

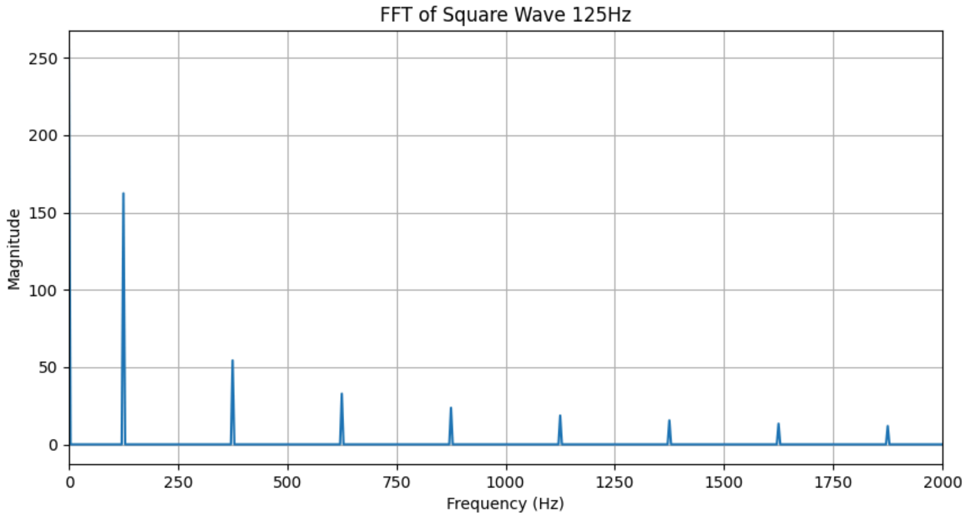
2.3.1. The FFT plot should display a strong peak at the fundamental frequency (≈15.625 Hz). Similar to the triangle wave, only odd harmonics should be present, but their amplitude decreases less rapidly (proportional to $\frac{1}{n}$).



(Square Wave FFT)

2.4. Square Wave 125Hz:

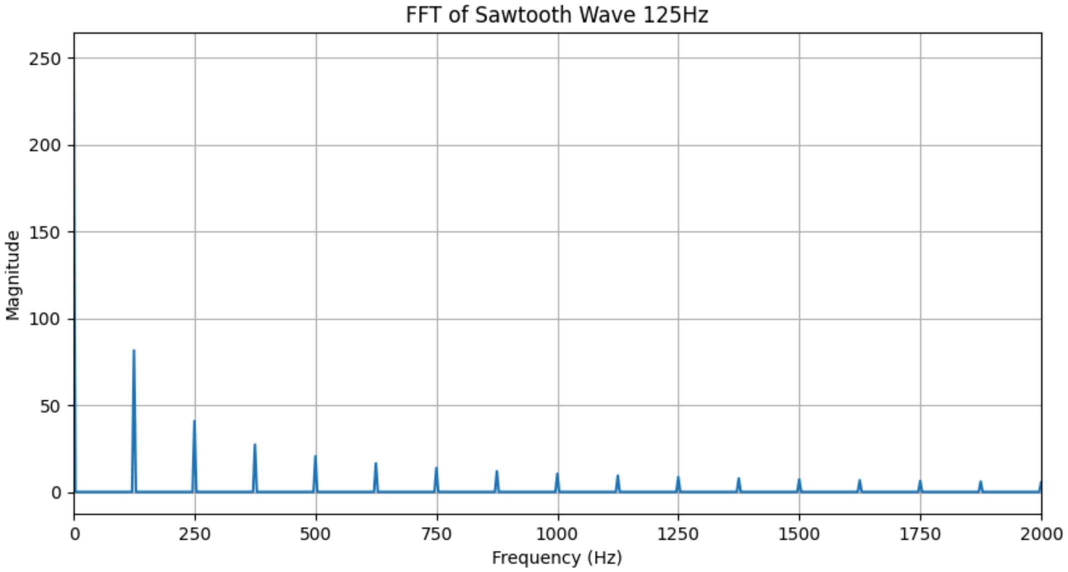
2.4.1. The FFT plot is expected to show a strong fundamental peak at 125 Hz and significant odd harmonics (375 Hz, 625 Hz, etc.), with amplitudes decreasing proportionally to $\frac{1}{n}$).



(Square Wave 125 Hz FFT)

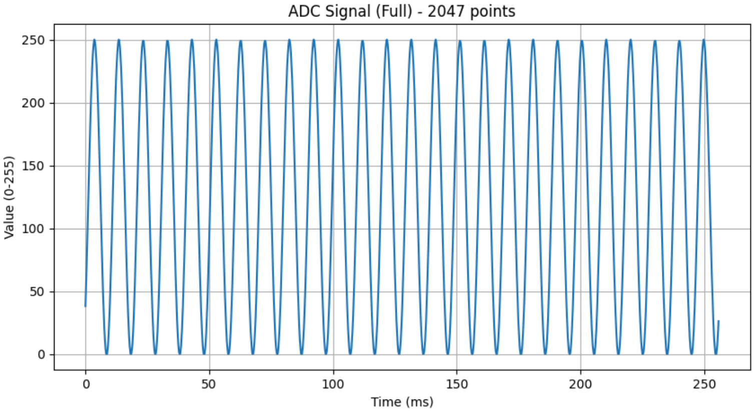
2.5. Sawtooth Wave 125Hz:

2.5.1. The FFT plot should show a strong fundamental peak at 125 Hz and significant peaks at both even and odd harmonics (250 Hz, 375 Hz, 500 Hz, etc.), with amplitudes decreasing with frequency.

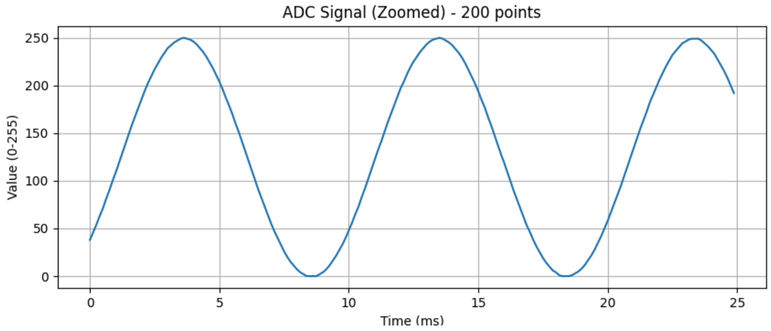


(Sawtooth Wave 125 Hz FFT)

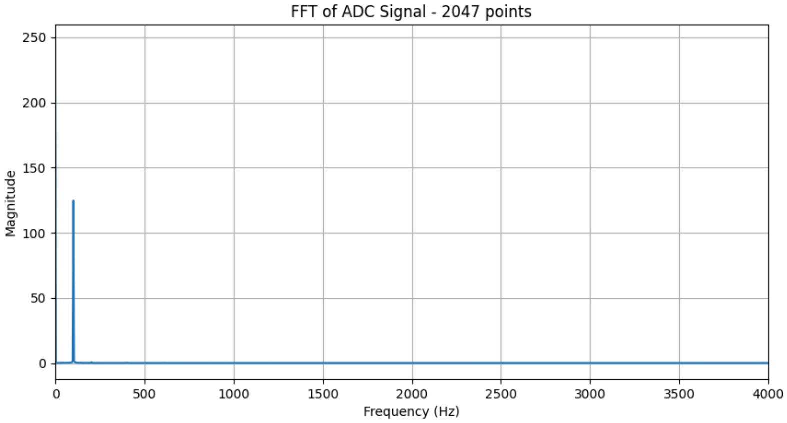
ADC Output Intake Generation



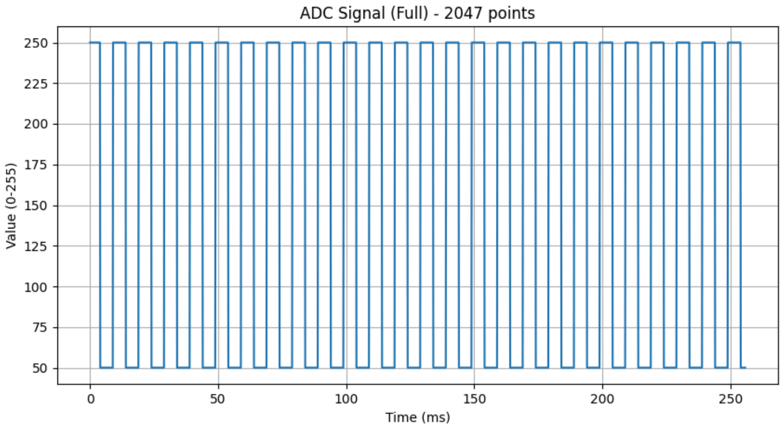
Sinewave (AWAVE100S.cmd and AWAVE100S.cmw for 100Hz) full plot



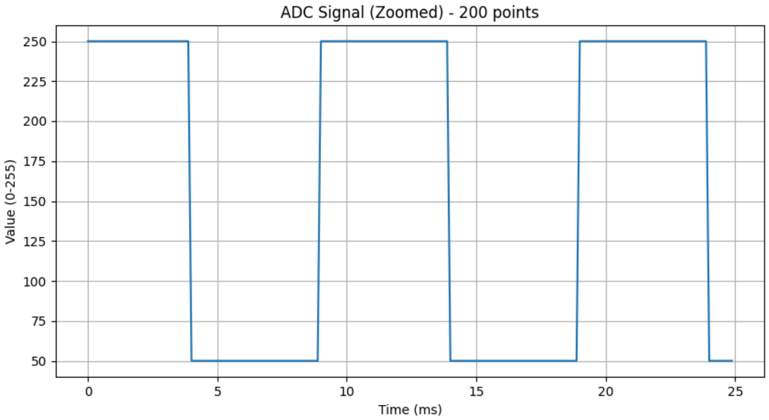
Sinewave (AWAVE100S.cmd and AWAVE100S.cmw for 100Hz) 200 points plot



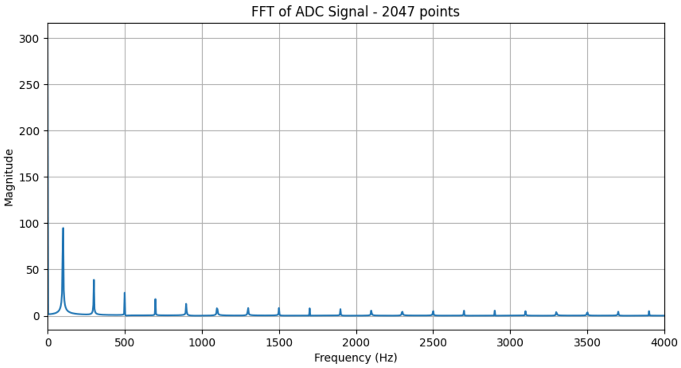
Sinewave (AWAVE100S.cmd and AWAVE100S.cmw for 100Hz) FFT plot



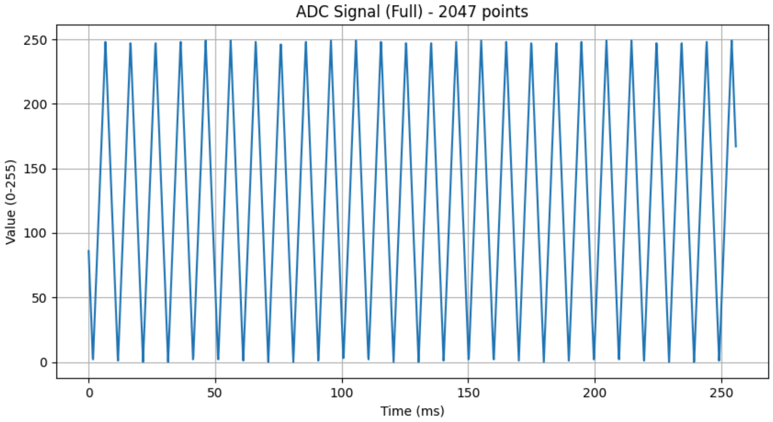
Square wave (AWAVE100Q.cmd and AWAVE100Q.cmw for 100Hz) full plot



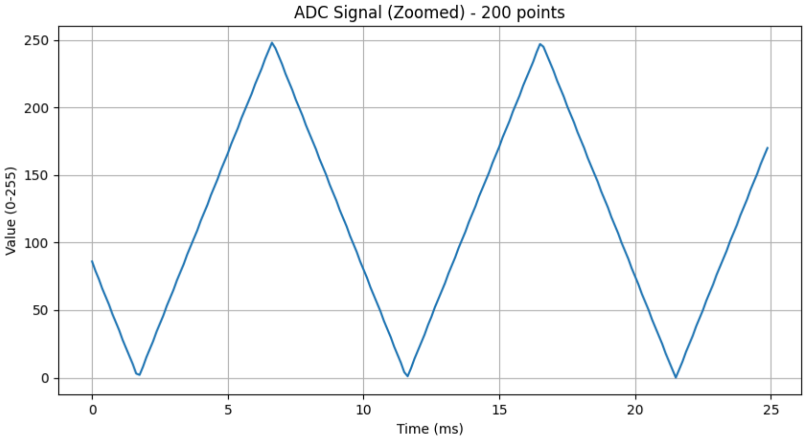
Square wave (AWAVE100Q.cmd and AWAVE100Q.cmw for 100Hz) Zoomed plot



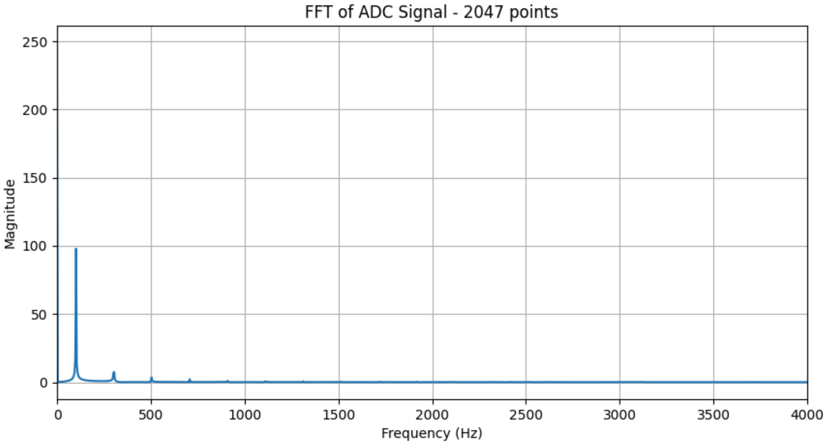
Square wave (AWAVE100Q.cmd and AWAVE100Q.cmw for 100Hz) FFT plot



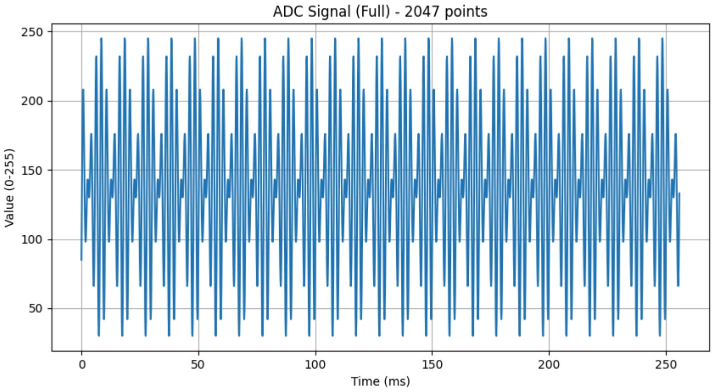
Triangle wave (AWAVE100T.cmd and AWAVE100T.cmw for 100Hz) Full plot



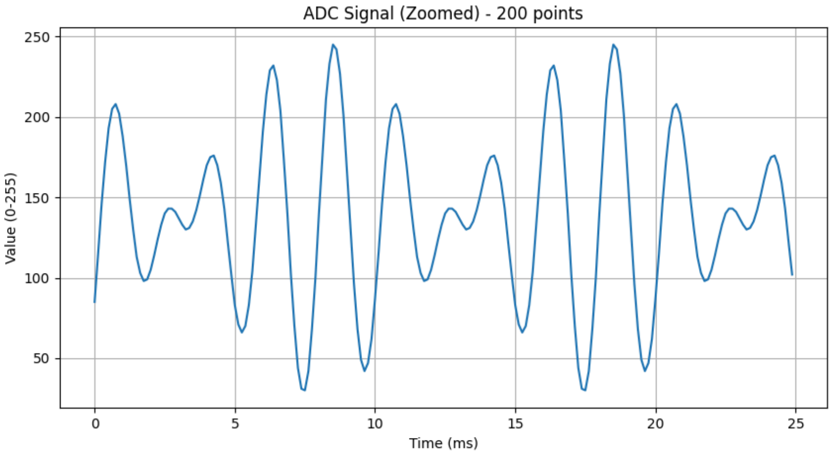
Triangle wave (AWAVE100T.cmd and AWAVE100T.cmw for 100Hz) Zoomed plot



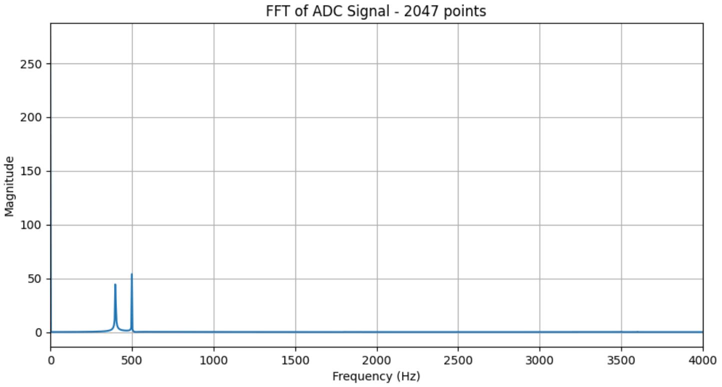
Triangle wave (AWAVE100T.cmd and AWAVE100T.cmw for 100Hz) FFT plot



Mixed Sine wave (AWAVE200S.cmd and AWAVE200S.cmw for ?Hz and ??Hz) Full plot



Mixed Sine wave (AWAVE200S.cmd and AWAVE200S.cmw for ?Hz and ??Hz) Zoomed plot



Mixed Sine wave (AWAVE200S.cmd and AWAVE200S.cmw for ?Hz and ??Hz) FFT plot

---

Conclusion:

The HCS12 assembly program utilized a timer 6 output compare interrupts to generate five different periodic waveforms (Sawtooth, Triangle, Square, and 125 Hz versions of Sawtooth and Square) with the intended shapes and fundamental frequencies. The program also outputted 2048 data points for each waveform via the SCI. The Python analysis script then parsed the SCI output data and generated accurate time-domain plots visualizing the complete waveforms. Furthermore, the FFT analysis performed by the script confirmed the expected frequency characteristics of each waveform type, including the presence and relative strength of their harmonics, consistent with Fourier theory.